Determining the Training Window for Small Sample Size Classification with Concept Drift

Indrė Žliobaitė
Faculty of Mathematics and Informatics
Vilnius University
Naugarduko St. 24, Vilnius, LT-03225, Lithuania
e-mail indre.zliobaite@mif.stud.vu.lt

L. I. Kuncheva
School of Computer Science
Bangor University
Dean Street, Bangor, Gwynedd LL57 1UT, United Kingdom
e-mail l.i.kuncheva@bangor.ac.uk

Abstract

We consider classification of sequential data in the presence of frequent and abrupt concept changes. The current practice is to use the data after the change to train a new classifier. However, if the window with the new data is too small, the classifier will be undertrained and hence less accurate that the 'old' classifier. Here we propose a method (called WR) for resizing the training window after detecting a concept change. Experiments with synthetic and real data demonstrate the advantages of WR* over other window resizing methods.*

1 Introduction

Classification of sequential data is often impaired by concept change. Classification of non-stationary streaming data is tightly related to network security, network traffic monitoring, navigation (robotics), surveillance systems and more. The adaptation of the classifier to the changing environments can be controlled by the size of a moving *training window* containing the latest N observations. Larger training windows are preferred for stationary distributions, while shorter windows are preferred after a sudden concept change. If the distribution is stationary, the training window should be allowed to expand up to a sufficient pre-defined size. An optimal size N^* should be determined online. The training window in this case may include data coming after the change as well as past data. The reason for this is that using only the data after the change, when only few observations are available, may produce an undertrained and hence inaccurate classifier. On the other hand, the properties of the data before the change may carry over the change point, and the old classifier may still be useful to a certain extent.

Methods for choosing the window size rely mostly on heuristics. More importantly, they do not make a clear-cut difference between the window size for *detecting* the change and the window size for *training* the classifier. There are two main approaches to handling variable window size. First, an explicit change detection is followed by a procedure to determine the size of the new training window [9, 6, 11, 3, 12]. The type of the change is identified as either abrupt or gradual, and a respective recipe is applied for resizing the window. If the change is deemed gradual, smaller part of the old data is cut off. Alternatively, if the change is deemed abrupt, a small, supposedly sufficient, training set is retained containing only the latest observations. The window resizing is guided by heuristics, and little attention is paid to the fact that the window needed for training the classifier and the window used for change detection should be considered separately.

The second approach to resizing the training window is based upon constant monitoring of the classification error, always assuming that there might have been a change in the past. A backward search is launched at each new observation (or batch thereof) in order to detect a past change point [7, 8, 10, 4, 1]. While Klinkenberg [8] chooses the new window by directly estimating its classification accuracy, the other detection methods only determine the possible change point. There is no recommendation of what the *t*raining window should be. It is thereby assumed that the amount of data coming after the change is sufficient for training the new classifier.

This study presents a method for choosing the *training* window size for classification of sequential data. The method is tailored for small sample sizes (frequent abrupt changes), where the classification tasks are relatively complex (high dimensionality, low separability). The method gives the optimal window size for two equiprobable multivariate Gaussian classes where the known change consists in a shift of the means. For comparison we chose three explicit window resizing methods that were the closest to the one proposed here: Drift detection (GAM) [6][1], a window selection algorithm for batch data (KLI) [8] and the Adaptive Windowing Algorithm by Bifet and Gavaldà (BIF) [4].

2 The Window Resize Method

2.1 Problem setup

Consider the following classification scenario. A sequence of i.i.d. data comes from source S_1. At time t_D a sudden concept shift occurs, in which source S_1 is replaced by source S_2. An online classifier model is trained progressively on the data from S_1 by expanding the training window with each new observation. At t_D the trained classifier becomes obsolete and should be replaced by a new classifier trained on S_2. Let C_1 be the classifier trained on the data from S_1, and C_2 be the classifier trained on the data from S_2. Since the data comes in a sequence, it would be in deficit straight after the change, and the newly trained C_2 will have erratic performance. On the other hand, if S_1 and S_2 are similar, the old classifier may still be more accurate than the new classifier until a sufficient training window of data coming from S_2 is accumulated.

We are interested in finding an optimal training data window for time point t. For that we need to detect the change point t_D and decide when classifier C_1 needs to be replaced by C_2 so as to minimize the generalisation error. Figure 1 illustrates the problem. Let $E^N(C)$ be the error rate of classifier C trained on N observa-

[1]We abbreviate the methods by the first three letters of the surname of the first author. The method that we propose will be referred to as Window Resizing and abbreviated as WR. We also experimented with FLORA [14] but the results were not competitive and we do not show them here.

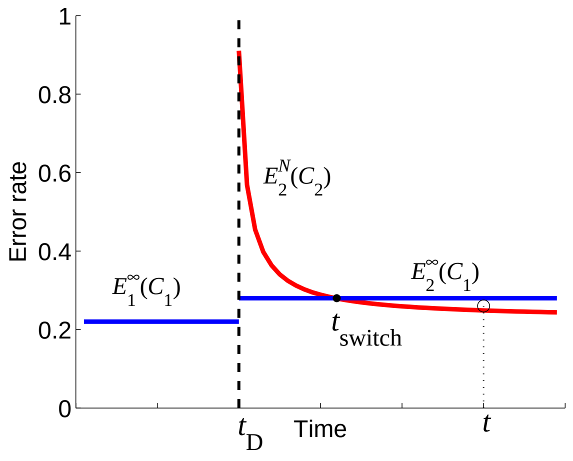

Figure 1. Error rates of C_1 and C_2. The time of the concept shift, t_D, is indicated by a vertical dashed line.

tions. Denote by $E^\infty(C)$ the asymptotic error rate of C obtained as $E^\infty(C) = \lim_{N\to\infty} E^N(C)$. Let $E_i(C_j)$ denote the error incurred by classifier C_j trained on data from source S_j with regard to the probability distributions in source S_i, where $i = 1, 2$. Shown in Figure 1 are the error rates of C_1 and C_2, the concept drift point t_D and the point of classification decision t. At t we should be using C_2 because its error rate $E_2^N(C_2)$ is smaller than the error rate of the old classifier C_1 i.e., $E_2^N(C_2) < E_2^\infty(C_1)$. It should be noted that the 'paired learners' method of Bach and Maloof [2] also examines the estimated accuracies $E_2^N(C_2)$ and $E_2^\infty(C_1)$ and makes a decision in favour of C_2 when the above inequality holds.

2.2 The optimal switch point for two Gaussian classes

Fukunaga and Hayes [5] show that, for parametric classifiers and two Gaussian classes, the classification error rate can be expressed approximately as

$$E^N(C) \approx E^\infty(C) + \frac{1}{N} f(C), \qquad (1)$$

where $f(C)$ is a function that depends on the classifier type, but not on N. Assuming that large enough number of observations have been accumulated from source S_1 until the change time t_D, we get $E_i^D(C_1) \approx E_i^\infty(C_1)$, $i = 1, 2$. On the other hand, C_2 is only trained on the limited number of observations from S_2. To determine when C_2 will be sufficiently trained to replace C_1, we solve for N the equation $E_2^\infty(C_1) = E_2^N(C_2)$, where

$E_2^N(C_2)$ is given by (1). The optimal window size after the change is

$$N^* = \frac{f(C_2)}{E_2^{\infty}(C_1) - E_2^{\infty}(C_2)}. \qquad (2)$$

Variants of $f(C)$ for different classifiers are tabulated in [5]. The error values $E_i^{\infty}(C_i)$ can be derived for specific distributions and classifiers [13]. Consider two equiprobable Gaussian classes in $\Re^n$ with means μ_1 and μ_2, and equal covariance matrices Σ. Assume that the change is an instant (uncoordinated) shift of both class means occurring at some known time point t_D. Denote the class means before the change by $\mu_1^{(1)}$ and $\mu_2^{(1)}$, and after the change, by $\mu_1^{(2)}$ and $\mu_2^{(2)}$. The error of the Linear Discriminant Classifier (LDC) for C_2 trained and evaluated on data from S_2 is the Bayes error, and is calculated as [13]

$$E_2^{\infty}(C_2) = \Phi\left(-\frac{\delta^{(2)}}{2}\right), \qquad (3)$$

where Φ is the cumulative distribution function of the standard normal distribution. The error rate $E_2^{\infty}(C_1)$ can be derived using:

• the Mahalanobis distances before and after the change, $\delta^{(1)} = (\mu_1^{(1)} - \mu_2^{(1)})^T \Sigma^{-1} (\mu_1^{(1)} - \mu_2^{(1)})$ and $\delta^{(2)} = (\mu_1^{(2)} - \mu_2^{(2)})^T \Sigma^{-1} (\mu_1^{(2)} - \mu_2^{(2)})$, respectively, where Σ is the common covariance matrix for the classes,

• the magnitude and directions of the change $\Delta_1 = \mu_1^{(2)} - \mu_1^{(1)}$ and $\Delta_2 = \mu_2^{(2)} - \mu_2^{(1)}$,

• the vector with coefficients of the LDC trained on source $S_1, \mathbf{w}^T = (\mu_1^{(1)} - \mu_2^{(1)})^T \Sigma^{-1}$.

The error of the 'old' classifier on the 'new' distribution is

$$E_2^{\infty}(C_1) = \frac{1}{2}\left\{\Phi\left(-\frac{\mathbf{w}^T\Delta_1}{\delta^{(1)}} - \frac{\delta^{(1)}}{2}\right)\right.$$
$$\left. + \Phi\left(\frac{\mathbf{w}^T\Delta_2}{\delta^{(1)}} - \frac{\delta^{(1)}}{2}\right)\right\}. \qquad (4)$$

The function relating the sample size and the classification error for LDC for classifier C_2 trained on data source S_j is [5]

$$f(C_2) = \frac{1}{2\sqrt{2\pi}\delta^{(2)}}\left[\left(1 + \frac{(\delta^{(2)})^2}{4}\right)n - 1\right]$$
$$\times \ \exp\left(-\frac{(\delta^{(2)})^2}{8}\right). \qquad (5)$$

With (3), (4) and (5) in place, and with a known change point t_D, we can calculate the optimal window size N^* from (2). Denoting the **optimal switch point** by t_{switch}, we get $t_{\text{switch}} = t_D + N^*$. Thus, the optimal window size at t is

$$N(t) = \begin{cases} t, & \text{if } t < t_{\text{switch}}, \\ t - t_D + 1, & \text{if } t \geq t_{\text{switch}}. \end{cases} \qquad (6)$$

We propose to use this result even though the true distributions may not be Gaussian. The class means before and after the change are estimated from the data. If dimensionality permits, a sample covariance matrix can also be estimated and the linear discriminant classifier can be applied. Otherwise, we can resort to the Nearest Mean Classifier (NMC) which only requires the class means in order to operate. In the latter case δ becomes the Euclidean distance. An unbiased estimate of the squared Euclidean distance δ^2 can be derived as

$$\delta^2 = \hat{\delta^2} - n\left(\frac{1}{N_1} + \frac{1}{N_2}\right), \qquad (7)$$

where $\hat{\delta^2}$ is calculated from the *estimates* of the means, N_1 and N_2 are the sample sizes for classes 1 and 2, respectively, and n is the data dimensionality. The correction should be applied for both $(\delta^{(1)})^2$ and $(\delta^{(2)})^2$.

2.3 Change detection using the raw data

The estimation of the probabilities of error and the parameters of the distributions needed for evaluating (2) hinges upon an accurate estimate of the change point t_D.

We propose to use the raw data for the change detection. Suppose that we have a sequence of observations labelled in c classes. To estimate the likelihood of a change at time d, where $1 \leq d \leq t$, we assume that the class means migrate independently of one another. Then the probability that there is a change at time d is

$$P(\text{change}|d) = 1 - \prod_{k=1}^{c} P(\text{no change in } \mu_k|d),$$

where μ_k is the mean for class k, $k = 1, \ldots, c$. Given that the data lives in $\Re^n$, the value of $P(\text{no change in } \mu_k|d)$ can be estimated using the p-value of the Hotelling multivariate T^2-test. This test compares the means for class k before and after the hypothetical change at d. If we use the notation $p_k(d)$ as the p-value returned by the Hotelling T^2-test comparing class k samples before and after time moment d, the probability of change at d can be estimated as

$$P(\text{change}|d) = 1 - \prod_{i=1}^{c} p_k(d). \qquad (8)$$

```
WINDOW RESIZE ALGORITHM (WR*)

input: a sequence of labelled observations

  1. Run backward search to find the likelihood
     $P(\text{change}|j)$ for $j = 1, \dots, t$, using (8).

  2. Estimate the change point with maximum
     likelihood $t_D = \arg\max_{j=1}^{t} P(\text{change}|j)$.

  3. Use (6) to calculate $N^{WR^*} = N(t_D)$. (For
     WR, use $N^{WR} = t - t_D + 1$.)

output: window size $N^{WR^*}$ ($N^{WR}$).
```

Figure 2. The WR* Algorithm

2.4 The WR* method

The WR* method is shown in Figure 2.4. We propose to use the optimal window size taking the change point with the maximum likelihood. Using (8) and (6), the optimal window size N^{WR^*} is calculated as

$$t_D = \arg\max_{j=1}^{t} P(\text{change}|j) \tag{9}$$

$$N^{WR^*} = N^*(t_D). \tag{10}$$

For comparison we will also include a version of the proposed method, called WR, where we do not use N^* but take instead the whole sample after the change

$$N^{WR} = t - t_D + 1. \tag{11}$$

3 Experimental evaluation

3.1 Set-up

We compare WR* with the three methods chosen for comparison, as well as with WR. We added a control scenario where the window is kept growing with the data regardless of any change. The method is called "all history" and is abbreviated as ALL. Thus the set of competing window resizing methods is: WR*, WR, GAM, KLI, BIF and ALL. For each dataset, artificial or real, we ran 6 synchronised experiments, one for each window resizing variant. The synchronization ensured that, once collated, the same sequence of data was submitted to each method. The Nearest Mean Classifier was used in all the experiments. The experimental design was chosen to showcase the proposed method, e.g., using small datasets and complex learning tasks.

BIF works only for 1-dimensional data, e.g., the running error. For this method to be used for n-dimensional raw data, a separate window is maintained for each dimension. The parameters needed for evaluating the discriminant functions are calculated on the respective windows. For example, the n components of the cluster means in $\Re^n$ may be derived using different window sizes. This model reflects the fact that the features may change at different pace.

KLI, BIF, WR* and WR include complete backward search starting at the current observation whereas GAM limits the search to the latest detected change.

3.2 Artificial data

Three data sets were generated: Gaussian data, STAGGER data and the moving hyperplane data. Owing to their frequent use, all three of them have acquired the status of benchmark data in the literature on concept change. However, the exact implementation varies from one study to another. Our protocol was as follows.

With each data set, for each observation in the sequence, we generated a random set of 100 observations to serve as the testing set at the current time point. The same testing set was used for all online classification methods in order to enable statistical comparison between them via paired t-test. Let $E(t)$ be the error rate of the online NMC at time point t, evaluated on the respective bespoke testing set. As an overall measure of the performance of the NMC we took the average of the errors at $t = 1, \dots, t_{\text{end}}$, i.e., $E_{\text{total}} = \frac{1}{t_{\text{end}}} \sum_t E(t)$. Due to the random nature of the streaming data, we average E_{total} over 100 independent runs and report in the table that error, denoted $\bar{E}_{\text{total}}$.

Gaussian data. We generated two Gaussian classes in $\Re^7$ with identity covariance matrices and $\mu_1^{(1)} = (1, 0, \dots, 0)^T$, $\mu_2^{(1)} = (-1, 0, \dots, 0)^T$. A sudden change was simulated by shifting the means by $\Delta_1 = (1, 0.2, 0, \dots, 0)^T$ and $\Delta_2 = (0.8, -0.4, 0 \dots, 0)^T$, respectively, as shown in Figure 3. The results are presented in Table 1. Statistically significant differences from a paired t-test are indicated.

STAGGER data. (Used by Widmer et al [14]). Each data point is described by 3 features, each with three possible categories: size $\in$ {small, medium, large}, colour $\in$ {red, green, blue} and shape $\in$ {square, circular, triangular}. Three classification tasks were to be learned in a course of 120 points. From point 1 to point 40, the classes to be distinguished are [size = small AND colour = red] vs all other values; from 41 to 80, [colour = green OR shape = circular] vs all other

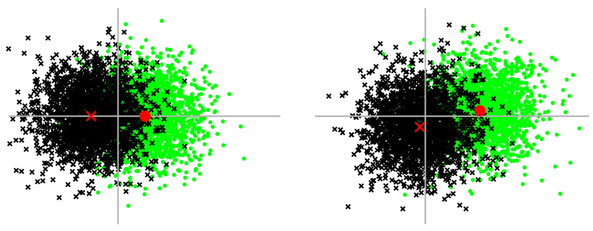

Figure 3. The first two features of the Gaussian data before and after the change

Table 1. Testing error $\bar{E}_{\text{total}}$ (in %) for the artificial data. The best accuracy for each column is underlined. The symbol next to the error rate indicates that the respective method is significantly: '•' worse than, '∘' better than, '−' no different to WR* ($\alpha = 0.05$).

Method	Gaussian	STAGGER	hyperplane
WR*	<u>17.63</u>	28.49	<u>11.05</u>
WR	18.33 •	<u>21.39</u> ∘	11.49 −
BIF	18.30 •	36.87 •	22.46 •
KLI	18.26 •	39.77 •	11.99 •
GAM	18.30 •	36.81 •	17.18 •
ALL	18.30 •	36.81 •	22.46 •

values; and from 81 to 120, [size = small OR size = large] vs all other values. Table 1 contains error $\bar{E}_{\text{total}}$.

Moving-hyperplane data. The data sequence is uniformly sampled from the unit square. The class labels are assigned according to a line through the centre of the square. The line rotates giving rise to change in the class descriptions (hence 'moving hyperplane'). Starting with a vertical discrimination line, we simulate 4 changes by positioning the discriminating line at 30°, 60°, 90° and 120°. To form a data stream, 50 i.i.d points were drawn from each source before the next rotation. The batch size for KLI was fixed at 12 which was empirically found to give KLI the best chance for this data set. Table 1 contains the results. Note that STAGGER and hyperplane data illustrate the case when the underlying assumptions for optimality of WR* do not hold.

3.3 Real data sets

We used 10 datasets from the UCI repository and simulated a concept change. With all the data sets, a cumulative error rate was maintained. The single error rate at time t, denoted $e(t)$, was estimated by testing the online classifier on the unseen data point coming at time t, before acquiring its class label ($e(t) = 0$ if correctly labelled, and $e(t) = 1$ if mislabelled). The cumulative error at time t is $E(t) = \frac{1}{t}\sum_{i=1}^{t} e(i)$. The final error $E(t_{\text{end}})$ was taken as the performance measure for the respective data set. In order to estimate statistical significance of the differences between the error rates of two methods, we used the McNemar test.

The data set that we chose are all 2-class problems with moderate size (100 -1000 instances) and dimensionality (10-100 attributes). To simulate concept change, we first permute the data and fix this sequence. Then we take the second half of the data and shift-rotate features from 1 to 5. Thus for the second half of the data original feature 1 is fed to the classifier as feature 2, original feature 2, as feature 3, and so on, while original feature 5 is submitted as feature 1. We used the same change pattern with all the datasets. Note that the feature values or the class labels are not changed in any way. The results are shown in Table 2. The differences between the errors of WR* and the other methods were not statistically significant, apart from BIF in 'cylinder' and GAM in 'SPECT heart', which were significantly worse than WR*. The 6 methods were ranked with respect to each data set, and the ranks were then averaged (shown in Table 1). WR* has the lowest rank by a large margin.

Sometimes there is an explicit suspected change point, e.g., due to change of operational circumstances, spatial location, or due to a time gap. For example, classification of network traffic may be affected by the release of a new software product at a particular (known) time; classification of customer preferences from retail records may face a concept change if a specialised store near by closes, etc. Thus, in addition to the experiments where the change point was unknown, we also tested the methods for a known change point. KLI and ALL gave the same result as with an the unknown change point. GAM, BIF and WR cut the window at the change point while WR* evaluated the data before and after the change point. Again, WR* has the lowest rank (bottom row in Table 1), which suggests that the success of WR* is due to the proposed window resizing calculation rather than to a clever change detection.

4 Conclusion

We propose a window resizing method for classification of sequential data. The data within the window is used for training the online classifier. We derive an expression for the optimal window size N^* for the case

Table 2. Testing error $\bar{E}_{\text{total}}$ (in %). The best methods for each data set are underlined.

Data set	WR*	KLI	WR	ALL	BIF	GAM
australian	35.34	$\underline{34.33}$	35.92	35.34	35.34	35.34
breast	$\underline{11.71}$	11.88	$\underline{11.71}$	$\underline{11.71}$	$\underline{11.71}$	$\underline{11.71}$
cylinder	$\underline{44.62}$	47.22	48.89	$\underline{44.62}$	48.89	$\underline{44.62}$
german	38.29	$\underline{37.89}$	38.29	38.59	38.59	38.59
Statlog heart	$\underline{38.48}$	39.22	$\underline{38.48}$	39.22	38.85	39.22
SPECT heart	26.50	$\underline{25.00}$	29.14	25.75	25.75	25.75
hepatitis	42.53	$\underline{36.69}$	42.53	43.18	43.18	43.18
ionosphere	25.86	$\underline{25.00}$	26.14	27.57	27.57	28.43
sonar	$\underline{37.92}$	41.30	$\underline{37.92}$	$\underline{37.92}$	$\underline{37.92}$	$\underline{37.92}$
vote	$\underline{11.18}$	12.10	11.64	11.87	11.87	11.87
rank	$\underline{2.60}$	3.20	3.50	3.80	3.95	3.95
rank (known change)	$\underline{2.70}$	3.45	3.95	3.00	3.95	3.95

of two Gaussian classes, the linear discriminant classifier (LDC) and change abrupt consisting of shift in the means. We proceed to propose a window resizing method using this result. The experimental results demonstrate that using the optimal window size makes all the difference in favour of the proposed method in comparison with three window resizing methods from the recent literature. The method is useful when the changes are moderate and the complexity of the data is high. In such cases a distinction between detection and training windows is particularly relevant because the old classifier is useful for longer time after the drift.

References

[1] R. P. Adams and D. J. C. MacKay. Bayesian online changepoint detection. University of Cambridge Technical Report, 2007.

[2] S. Bach and M. Maloof. Paired learners for concept drift. In *Proc. The Eighth IEEE International Conference on Data Mining ICDM'08*, pages 23–32, 2008.

[3] M. Baena-Garcia, J. del Campo-Avila, R. Fidalgo, A. Bifet, R. Gavalda, and R. Morales-Bueno. Early drift detection method. In *ECML PKDD 2006 Workshop on Knowledge Discovery from Data Streams, 18 Set 2006, Berlin, Germany*, 2006.

[4] A. Bifet and R. Gavalda. Learning from time-changing data with adaptive windowing. In *SDM*. SIAM, 2007.

[5] K. Fukunaga and R. R. Hayes. Estimation of classifier performance. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 11(10):1087–1101, 1989.

[6] J. Gama, P. Medas, G. Castillo, and P. P. Rodrigues. Learning with drift detection. In *Advances in Artificial Intelligence - SBIA 2004, 17th Brazilian Symposium on Artificial Intelligence, ser. Lecture Notes in Computer Science*, volume 3171, pages 286–295. Springer Verlag, 2004.

[7] D. Kifer, S. Ben-David, and J. Gehrke. Detecting change in data streams. In *Proceedings of the 30th International Conference on Very Large Data Bases (VLDB 2004)*, Toronto, Canada, 2004.

[8] R. Klinkenberg. Learning drifting concepts: Example selection vs. example weighting. *Intelligent Data Analysis*, 8(3):281–300, 2004.

[9] R. Klinkenberg and I. Renz. Adaptive information filtering: Learning drifting concepts. In *AAAI-98/ICML-98 workshop Learning for Text Categorization, Menlo Park,CA*, 1998.

[10] I. Koychev and R. Lothian. Tracking drifting concepts by time window optimization. In *Research and Development in Intelligent Systems XXII Proceedings of AI-2005, the Twenty-fifth SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence. Bramer, Max; Coenen, Frans; Allen, Tony (Eds.), Springer.*, 2005.

[11] M. Lazarescu, S. Venkatesh, and H. Bui. Using multiple windows to track concept drift. *Intelligent Data Analysis*, 8(1):29–59, 2004.

[12] K. Nishida and K. Yamauchi. Detecting concept drift using statistical testing. In V. Corruble, M. Takeda, and E. Suzuki, editors, *Discovery Science*, volume 4755 of *Lecture Notes in Computer Science*, pages 264–269. Springer, 2007.

[13] S. Raudys. *Statistical and neural classifiers: an integrated approach to design.* Springer-Verlag, London, UK, 2001.

[14] G. Widmer and M. Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996.